International Journal of Electrical and Computer Engineering (IJECE)
Vol. 7, No. 5, October 2017, pp. 2855~2862
ISSN: 2088-8708, DOI: 10.11591/ijece.v7i5.pp2855-2862

A Novel Optimization towards Higher Reliability in Predictive Modelling towards Code Reusability

Manoj H. M.[1], Nandakumar A. N.[2]
[1]Jain Univeristy, Bangalore, India
[2]Dept. of Computer Science & Engg, New Horizon College of Engg, Bangalore, India

Article Info

Article history:
Received Feb 21, 2017
Revised Jun 9, 2017
Accepted Sept 11, 2017

Keywords:
Code reusability
Object oriented programming
Optimization
Software engineering

ABSTRACT

Although, the area of software engineering has made a remarkable progress in last decade but there is less attention towards the concept of code reusability in this regards.Code reusability is a subset of Software Reusability which is one of the signature topics in software engineering. We review the existing system to find that there is no progress or availability of standard research approach toward code reusability being introduced in last decade. Hence, this paper introduced a predictive framework that is used for optimizing the performance of code reusability. For this purpose, we introduce a case study of near real-time challenge and involved it in our modelling. We apply neural network and Damped-Least square algorithm to perform optimization with a sole target to compute and ensure highest possible reliability. The study outcome of our model exhibits higher reliability and better computational response time

Corresponding Author:

Manoj H.M,
Research Scholar,
National Chung Cheng University,
ROC Jain Univeristy, Bangalore, India
Email: manoj.hmhm@gmail.com

1. INTRODUCTION

With the changing dynamics of the client's requirement, the methodologies adopted in software project development organization are consistent changing in order to keep a pace with current demands. Although, there are evolution of various updated tools, techniques, and technologies in Information Technology, but adoption of all these are certainly not cost effective in nature [1]. In this direction, code reusability plays a contributory role in introducing certain software metrics that significantly assists in upgrading the domain of software engineering [2-4]. At present, the conventional and frequently exercised techniques of code reusability are highly connected with the structured method of assessing and adjusting the object-oriented development process [5]. There is certain research work that has been focused on visualizing the benefits of code reusability with respect to object-oriented system [6-7]. With an aid of potential software engineering approaches, the significance of code reusability can be measured using certain standard metrics [8]. The biggest question here is that although there are so many past techniques for enhancing code reusability than why the adoption of it in the existing organization is so less. Taking a case study of smaller scale software development industry, it is quite well known that adoption of such quality programs never workout for them due to lack of or constraint of liquidity. This fact eventually tells us that progress towards code reusability is extremely less. So, the existing paper also reviews some of the recent techniques towards code reusability and checks the extent of progress and tradeoffs. Therefore, it can be said that if an appropriate knowledge about the advantageous points can be understood that it may significant help us in formulating a novel software reusable component in future and thereby bring into practice. The investigation conducted over this stream of software engineering recommends that it can potentially enhance the productivity system and offer significant degree of enrichment to the software design. In the area of software engineering, the term 'software reuse' is normally linked with the productivity efficient in order to enhance the processes involved in software development. A different form of the framework used for validation of the

appropriateness of design process highly depends over the preciseness of the outcome achieved by the framework. Hence, there is a need of investigation in two directions where the first direction should focus on evolving up with some novel model of code reusability while second direction should focus on incorporating optimization. The significant advantage towards such modeling is that it will be easy to be adopted and will be applicable over larger range of application that uses object oriented system. Fortunately, there is a bigger scope of enhancing the optimization technique with larger range of optimization algorithm being available in existing times. However, there is no much optimization being carried out towards code reusability system. There is also a possibility that existing approaches towards software metrics be further investigated and clubbed with experimental data to further investigate the better possibilities of code reusability and thereby a good chance for evolving up with a predictive system.

The proposed system therefore introduces a mechanism of optimization where the code reusability is emphasized by adoption a true case study and analytical approaches. We also emphasize on algorithm efficiency. Section 2 discusses about the research methodology of proposed system followed by elaborated discussion of algorithm implementation in Section 3. Section 4 briefs out the results being accomplished while summary of the paper are discussed in Section 5.

1.1. Background

This section discusses about the existing approaches that has been carried out towards code reusability. Our review work has already discussed about different techniques and trade-off in existing system [9], we add further update towards the research progress in this domain. First of all, there is significantly very less number of research progresses in the topic of code reusability from the year 2010 till date. However, we only discuss the related work in this perspective. Hudaib et al. [10] have presented a classification-based approach for software reusability using self-organizing map. Khoshkbarforoushha et al. [11] have discussed about software metric in order to incorporate reuable components in programming language. Tibermacine et al. [12] have addressed the constraints during modeling of software reuse on architectural design. Tahir et al. [12] have presented a reusability-based framework that offers efficient selection of components. The research towards of reusability was also seen in different pattern in most recent times. Vegt et al. [13] have presented such architecture that emphasizes on reusability of gaming components. The architecture has been shown to have very low dependability towards conventional software patterns and existing practice and protocols of coding. Demraoui et al. [14] have presented a synchroncity-based approach for enhancing the reusability performance over warehouses. The authors have used case-based reasoning approach in order to enhance the process of software reusability. Ahmaro et al. [15] have presented a discussion towards existing practical adoption of software reusability considering the case study of software development organization in Malaysia. The paper gives the fair idea of various forms of reusable approaches e.g. application product lines, design patterns, COTS integration, program generators, component-based development, configurable vertical applications, legacy system wrapping, aspect-oriented software development, etc. Efat et al. [16] have addressed the problems pertaining to component reuse and analyzed the problem using data repository. The author commented that in order to address the problem of reusability, the cost for space-time should be addressed for the software components. The technique has also introduced two different algorithms where one focuses on selection of attributes while other focuses on ranking as well as prioritizing it. The study outcomes were assessed using success and failures cases of test outcomes. Mojica et al. [17] have presented an empirical-based investigation towards software reusability. Mobile applications are considered as a case study where the authors investigated software reusability. The author also commented that design of the mobile applicatons, at present; make use of various forms of classes, inheritance, and library reuse. It will significant assists the developer to further upgrade the performance of software reusability. Nazir et al. [18] have presented a discussion towards selection of software components by introducing a unique analytical framework considering case study. The author basically uses an existing mechanism and then modifies into analytical network process that significantly depends on feedback (i.e. expert opinion) as well as dynamic structure of the network. The significant advantage of this study is its supportability of various types of reusable factors. It also assists in decision making for complex networks. The study outcome is assessed using graphical weights. Spoelstra et al. [19] have discussed the usage of the software reuse towards the organization that practices agile methodologies. The authors have also discussed about various reuse factors on existing models as well as discussed about validated outcome of the study. Basically, the study represents a type of software management tool to support reusbability concept. Zhu et al. [20] have discussed about the software reusability in terms of framework reusability with respect to distributed process of simulation. The solution provided by the author is a framework with reusable component that is constructed by the domain experts. The author also introduces multiple service-oriented interfaces for performance enhancement. The combined work of Manoj and Nandakumar [21] have described code reusability concept for cost optimization in IT system. The study

provides an analytical model to bring relationship among the standard metric components and code reusability. The Markov mode is utilized for significant information extraction which provides highest code reusability value.

1.2. The Problem

The prior section has presented a brief discussion of some of the related work connected with code reusability. Code reusability is one subset of software reusability which has higher rate of technical adoption by many software engineers irrespective of the scale of the organization. However, inspite of such massive usage of code reusability, there is a very less number of attentions being attracted from the research community towards this topic. Hence, there are various set of open-end problems that are yet to be addressed in future.

Following are the open research problems after reviewing the existing literatures:-

- There is no solid research conducted for code reusability in software engineering till date.
- Majority of the existing research lacks model validation for which reason the study outcomes of existing system cannot be directly said to be applicable.
- There is no availability of any benchmark models or techniques towards code reusability or optimization technqiues which makes its still an open research issues.
- None of the existing technqiues are found to adopt any technique that offers algorithm cost effectiveness.

Therefore, after reviewing the existing problems, it can be concluded that there is a need to work on such above mentioned research problems. Hence, the problem statement of the proposed study is –"It is significantly a challenging task to formulate a system model that applies optimization in order to enhance the predictiveness of the code reusability." The next section discusses about the proposed system to address such issues.

1.3. The Proposed Solution

The proposed system discussed in this paper is a continuation of our prior work [22]. The prime purpose of this work is to introduce a framework where code reusability can be tested and analyzed with varied ranges of near real-time problems associated with software development. The schematic architecture of the proposed system is as shown in Figure 1.

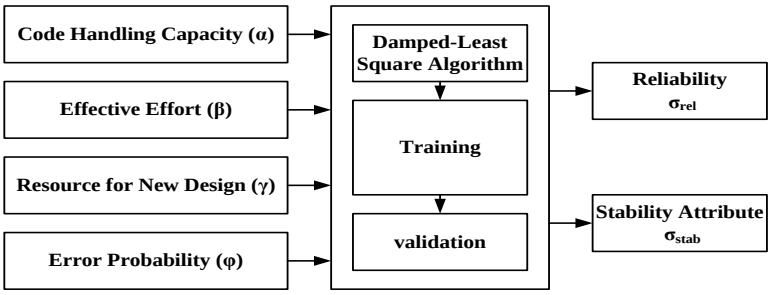


Figure 1. Schematic architecture of proposed system

There are mainly four essential stages involved in designing proposed framework e.g. i) in the first stage, we take sample object-oriented source code from two software projects and extracts its software metric values with respect to Coupling between objects, Response for classes, weighted methods per class, Depth of inheritance tree, Number of Children, ii) in the second stage, we introduce 4 different problems in software development as case study, iii) in the third stage, we compute code reusability using concept mentioned in [22], and iv) we apply optimization technique to ensure reliability of the outcome obtained for code reusability. This part of the study formulates an empirical approach for further computing the coding attribute,

$$C_A = \frac{1}{R}(P-Q) \quad (1)$$

Where, the variable P represents product of software metrics, total effort, and hours of work. The variable Q represents product of software metrics, effort for incorporating code reusability, and hours of work, while R will represent difference of cumulative effort days with days required to incorporate code reusability. We also formulate condition where software engineers have to deliver the software code with inclusion of certain range of code reusability to ensure cost saving of production in future. This predicted outcome of code reusability is then assessed using a machine learning approach to ensure higher degree of reliability as well as stability in the outcome of proposed system. The next section discusses about research methodology adopted in proposed study.

2. RESEACH METHODOLOGY

The design of the proposed research work is carried out considering analytical research methodology for enhancing code resuabilty in software engineering. As optimization is the prime motive of proposed study therefore a machine learning approach is applied for this purpose. This job is done by using neural network for the purpose of ranking optimization in software engineering. The prime reason for adopting neural network for optimization is because of its multi-tasking capabilities. We construct a schematic architecture that takes the input of 4 different types of problems associated with real-time software development that finally results in computation of Reliability as well as stability attribute. Following are brief description of the inputs towards the model for code reusability:

- **Code Handling Capacity (α)**: This attribute is considered as the maximum number of software codes that can be newly designed and develired by a team for one month against the client's requirement.
    - *Rationale*: This attribute α will directly represent the quantification of optimization technique toward proposed code reusability concept. Increase in the value of this attribute by keeping the other attributes similar is the direct indicator of optimization.
- **Effective Effort (β)**: This attribute is considered as the precise time consumed by desiginer when a code is developed by incorporating code reusability within it.
    - *Rationale*: This attribute β will represent saving of production time when optimization is performed to see how much time is required in i) either building the code from scratch or ii) applying the code reusability concept.
- **Resource for New Design (γ)**: This attribute defines number of resources being utilized for developing the new code with code reusability within it.
    - *Rationale*: This attribute γ will directly represent the cost involved in production. The developer is assumed to design the new code in such a way that it should have certain thresholded percentage of code reusability. Hence, this value should be kept as low as possible.
- **Error Probability (φ)**: In order to check the internal validaity of the proposed model, we intentionally incorporate certain range of error as an uncertainty.
    - *Rationale*: The entite concept of code reusability and its optimization is framed without even knowing the form of requirement of client. We call this factor as *uncertainty*. Therefore, this attribute φ will perform computation of code reusability by inclusion of uncertainty factor or error probability.

We use all the above mentioned attributes as input towards the processor, which upon processing will give the output of reliability and stability. We apply Damped-Least Square algorithm as the optimization algorithm in order to perform optimization using neural network.

3. ALGORITHM IMPLEMENTATION

The implementation of the proposed algorithm is carried out using Matlab. The algorithm design mainly focuses on improving the study outcomes of optimization i.e. reliability and the stability. The algorithm implications initiate with the 4 different forms of the input i.e. Code Handling Capacity (α), Effective Effort (β), Resource for New Design (γ), and Error Probability (φ). The numerical values of this input are obtained by considering 2 software engineers capable of deliverying 2 and 3 software codes for new projects in 26 working days. Total of 8 hours of working time is considered for this purpose. We adhere to the standard of the neural network based optimization policy by associating weights with the input attributes. The cardinality of the input layer is 4 while that of the intermediate layer is 24 and 2. Similarly, the cardinality of the outcome is 2.

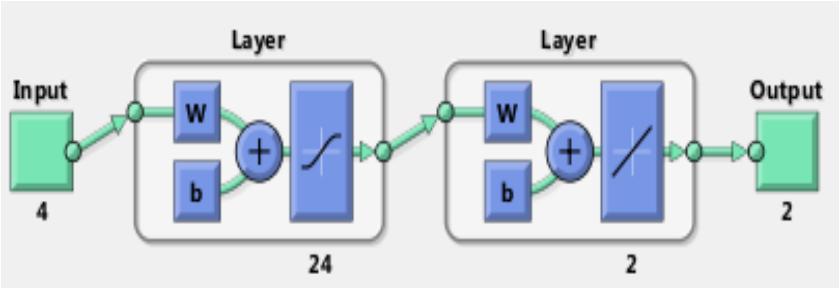


Figure 2 Schema of neural network used in proposed system

The neural network mapping config of nodes are 4(input)-24(intermediate)-2(ouput) as shown in Figure 2, where cardinality of input nodes is 4, cardinality of hidden nodes is 24, and cardinality of output node is 2.The essential steps involved in proposed algorithm are as follows:

- **Optimization Algorithm for Code Reusability**

Input: Code Handling Capacity (α), Effective Effort (β), Resource for New Design (γ), and Error Probability (φ).
Output: σ_{rel} (reliability),σ_{stab}. (stability attribute)
Start:
1. init Δ=[α, β, γ, ϕ]
2. Compute weight (δ)
3. a=min-max(value(Δ))
4. *norm*(Δ, o)→[(-4, +4)(0.1, 0.9)]
5. Apply traing_Alg
6. Error→MinMax(N_o-T_o)
7. Compute $\sigma_{rel} \Rightarrow \sum c_1.(\Delta - c_2).\delta$ & σ_{stab}.

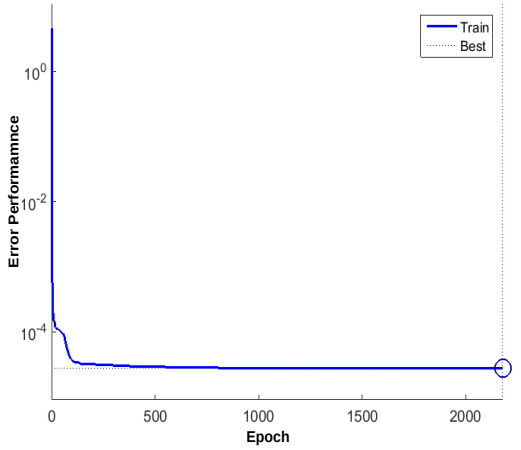
End

The above mentioned algorithm adopts the design principle of Figure 1 and performs initialization of multiple attributes associated with the case studies i.e. (α, β, γ, φ). After initializing the input attributes (Line-1), the next important step of algorithm is to compute weight as {(4(input layer)x24(hidden layer))+(24(hidden layer)x2(output layer))=144}. This weight is used for computing output (Line-7). The prime target of this algorithm is to evolve up with a number of perceptrons layers for the purpose of predicting reliability attribute and stability attribute. Various forms of iterations have been observed in order to look for an ellite outcome of optimization. The algorithm performs processing considering various numbers of permutations with input attributes. Apart from this data, we also constructed a training dataset where elongated series of training data associated with our case study is developed. This data was called during implication of the training algorithm. We split the data into two parts where the bigger proportion of the data is considered for applying itself to training operation in neural network and smaller proportion of the data is considered for carrying out internal model validation. Both the inputs attribute Δ and output attribute σ has been subjected to normalization array of (0, +1). The numerical values of input attributes Δ is obtained and min-max algorithm is applied (Line-3). The algorithm also performs normalization of the both input and output numerical outcomes to ensure that the numerical outcomes are statistically within probability limits (Line-4). For easiness in computation, we choose the normalization value of input as -4 to +4 while that of output is considered to be 0.1 to 0.9. These values can be changed according to the test environment; however, we choose this value of sharp resemblems with out consideration of case study. A feedforward neural network is used as a media of training algorithm (Line-5) followed by computation of output from neural network N_o and training T_o. We find difference of these twor intermediate outcomes and again apply min-max algorithm in order to extract the possible error (Line-6). This error is used for internal model validation in later stage of model checking. Finally, we compute the reliability attribute σ_{rel} whose dependable variables are c_1, c_2, Δ, and δ (Line-7). The first variable c_1 represents a higher-limit constant 1/0.8 while the second variable c_2 represent lower-limit constant of value 0.1. These values can be suitably changed based on the necessity of the anticipated convergence outcomes. The considerations of these values are purely based on probability theory. Adoption of higher limit 0.8 is done as statistically, the permissible higher limit is till 0.8, while numbers more than it is considered as impractical. The last variable δ is weight which is equivalent to 144 in our case. Hence, once the reliability attribute is computed, we check for stability factor i.e. σ_{stab} just to ensure the total number of occurance of reliable outcomes. If the total frequencies of majority of the

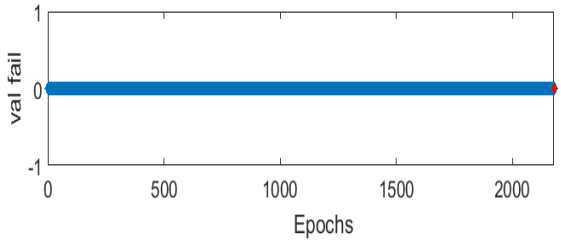
generated values of σ_{rel} is nearly similar than we consider that the proposed test scenario is highly stable. The training performance was checked for 1000-5000 iterations in order to see the convergence performance of both experimental and hypothetical outcomes. An interesting fact about this algorithm is that it is more prone to converge towards low error rate as an ellite outcome, which ensures that proposed system offers and outcome whose validation leads to highly authenticated outcome. This also means that proposed system can truelly optimize the concept of code reusability under consideration of various real-life challenges in software project development. The performance of the proposed algorithm is nearly found similar under different forms of changes being carried out towards testing the algorithm performance. At the same time the complexity of the algorithm is also found quite low. The next section discusses about the outcomes accomplished from proposed study.

4. RESULT AND DISCUSSION

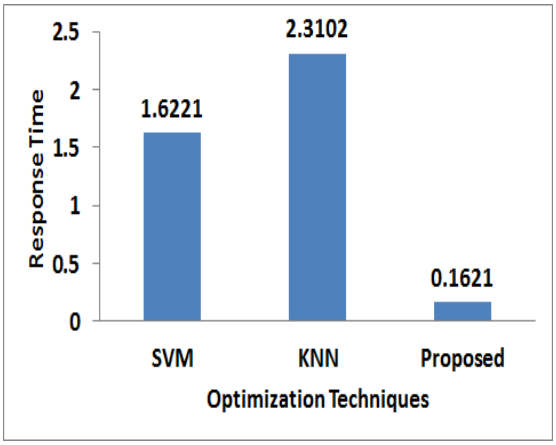
As the proposed study focuses mainly on two factors i.e. optimization and reliability, we consider testifying the study outcome of the proposed system on the basis of error performance of proposed model, validation performance, and response time mainly. The model is allowed to iterate for 20,000 rounds in order to check the convergence performance. Figure 3 (a) shows the trained result significantly converges with the best fit result thereby showing the higher reliability of the proposed model. The model reliability is further proven by linearity trend of the validation curve in Figure 3 (b). Although, 20,000 epochs has been given, but the convergence towards elite outcome was seen only within 2178 epoch.



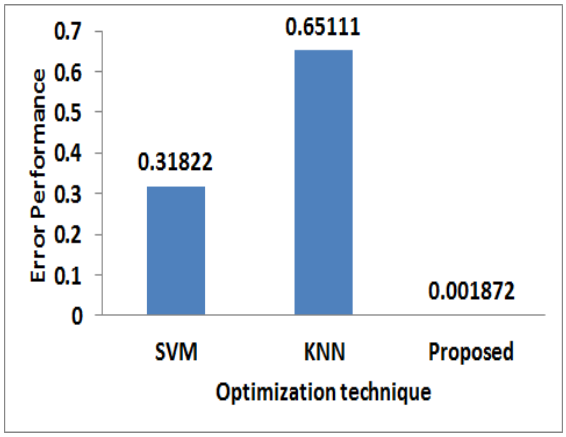
(a) Error performance



(b) Validation performance



(c) Comparative response time



(d) Comparative error performance

Figure 3. Outcome of proposed study

As there are various forms of optimization techniques therefore we compare our optimization technique using neural network and various others frequently used optimization technqiues e.g. k-Nearest Neighbour (KNN) and Support Vector Machine (SVM) under similar environment parameters. The analysis shows that proposed study offer significantly lower response time (Figure 3. (c)) and higher reliability as seen from error performance (Figure 3. (d)). KNN-algorithm doesn't give better outcome as it is much distance-based approach for all the training data while performing optimization. Therefore response time is quite higher as well as error is also too high. On the other hand, SVM has better performance in contrast to KNN due to its ability to perform both linear and non-linear classification. Adoption of kernel-based approach also assists it to offer better error performance. However, the response time could be only lowered to certain extent in SVM, which is not cost effective in nature although, it offer lowered error. The proposed system uses neural network whose accuracy level can be increasingly controlled in every cycle of epoch. On the other hand, adoption of damped least square algorithm significantly assists in solving non-linear optimization problem and hence the contribution of neural network is proposed system is more inclined to error performance than on performance time. Hence, it can be said that proposed system offers a cost effective as well as highly reliable optimization of the framework that supports code reusability.

5. CONCLUSION

The concept of software reusability has received lots of attention within research community 10 years back that resultsi n evolution of various standard software metrics. However, there is less number of interests towards this domain found as there are very less research papers in this. Code reusability is one of the part of software reusability that has never being investigated in past although the concept of code reusability is practiced in many organization without even following any protocols. The prime reason behind it is that there are no such benchmarked models in this regards. Therefore, we address this problem by taking a case study that mimicks real-time problem in software development. We also use sample software projects and computes its conventional software metrics values that we use for our analytical modelling. We introduce a formulation to compute the code attribute with an inclusion of code reusability logic. Neural network is applied for optimization to find that proposed system offer extremely lower error score and reduced computational time in comparison to existing optimization techniques.

REFERENCES

[1] J. Portman, "Building Services Engineering: After Design, During Construction," *John Wiley & Sons*, 2016
[2] R. Lutowski, "Software Requirements: Encapsulation, Quality, and Reuse," *CRC Press*, 2016
[3] D. Wiebusch, "Reusability for Intelligent Realtime Interactive Systems," *BoD – Books on Demand-Computer*, 2016
[4] J. Sametinger, "Software Engineering with Reusable Components," *Springer Science & Business Media*, 2013
[5] M. Kraeling, Andrew McKay, "Software Engineering for Embedded Systems," *Elsevier Inc*, 2013
[6] L. Antovski1 and Florinda Imeri2, "Review of Software Reuse Processes," *International Journal of Computer Science Issues*, vol. 10, issue 6, no. 2, 2013
[7] P. S. Sandhu, Aashima, P. Kakkar and S. Sharma, "A survey on Software Reusability," *IEEE-International Conference on Mechanical and Electrical Technology, Singapore*, pp. 769-773, 2010
[8] N. Padhy, R. Panigrahi and S. Baboo, "A Systematic Literature Review of an Object Oriented Metric: Reusability," *IEEE- International Conference on Computational Intelligence and Networks, Bhubaneshwar*, pp. 190-191, 2015
[9] H.M. Manoj and A.N. Nandakumar, "A Survey on Modelling of Software Metrics for Ranking Code Reusability in Object Oriented Design Stage," *International Journal of Engineering Research & Technology (IJERT)*, vol. 3, issue. 12, 2014
[10] A. Hudaib, A. Huneiti, I. Othman, "Software Reusability Classification and Predication Using Self-Organizing Map (SOM)," *Communications and Network*, pp. 179-192, 2016
[11] A. Khoshkbarforoushha, P. Jamshidi, M. F. Gholami, L. Wang and R. Ranjan, "Metrics for BPEL Process Reusability Analysis in a Workflow System," *in IEEE Systems Journal*, vol. 10, no. 1, pp. 36-45, March 2016.
[12] C. Tibermacine, S. Sadou, M.T.T.That, and C.Dony, "Software Architecture Constraint Reuse-by-composition," *Future Generation Computer Systems*, vol. 61, pp.37-53, 2016.
[13] M. Tahir, F. Khan, M. Babar, F. Arif, and F. Khan, "Framework for Better Reusability in Component Based Software Engineering," *The Journal of Applied Environmental and Biological Sciences (JAEBS)*, vol. 6, pp.77-81, 2016
[14] W.V.D. Vegt, W. Wim, E. Nyamsuren, A. Georgiev, and I.M. Ortiz, "RAGE Architecture for Reusable Serious Gaming Technology Components," *International Journal of Computer Games Technology*, vol. 3, 2016
[15] L. Demraoui, H. Behja, and R. B. Abbou, "A Case-based Reasoning Approach to the Reusability of CWM Metadata," *In Systems of Collaboration (SysCo), International Conference*, pp. 1-6, 2016
[16] I.Y.Y. Ahmaro, M. Z. b. M. Yusoff, and A. M. Abualkishik, "The Current Practices of Software Reusability Approaches in Malaysia," *In Software Engineering Conference (MySEC), 8th Malaysian*, pp. 172-176, 2014.
[17] M. I. A. Efat, M. S. Siddik, M. Shoyaib, and S. M. Khaled, "Feature Prioritization for Analyzing and Enhancing Software Reusability," *In Informatics, Electronics & Vision (ICIEV), International Conference*, pp. 1-5, 2014

[18] I. J. Mojica, B. Adams, M. Nagappan, S.Dienst, T. Berger, and A. E. Hassan, "A Large-scale Empirical Study on Software Reuse In Mobile Apps", *IEEE software,* vol. 31, no. 2, pp.78-86, 2014.
[19] S. Nazir, S. Anwar, S. A. Khan, S. Shahzad, M. Ali, R. Amin, M. Nawaz, P. Lazaridis, and J. Cosmas, "Software Component Selection Based on Quality Criteria Using the Analytic Network Process," *In Abstract and Applied Analysis,* vol. 2014
[20] W. Spoelstra, M. Iacob, and M. V. Sinderen, "Software Reuse in Agile Development Organizations: a Conceptual Management Tool," *In Proceedings of the ACM Symposium on Applied Computing*, pp. 315-322, 2011
[21] Manoj H. M, Dr. Nandakumar A.N, " Constructing Relationship Between Software Metrics and Code Reusability in Object Oriented Design," (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 7, No. 2, 2016
[22] F. Zhu, Y. Yao, H. Chen, and F. Yao, "Reusable Component Model Development Approach for Parallel and Distributed Simulation," *The Scientific World Journal*, pp. 12, 2014

BIOGRAPHIES OF AUTHOR

Manoj H M, Currently working as Assistant Professor, Dept of CSE, and Don Bosco Institute of Technology, Bangalore, India. He has total experience of 6 years & 4 months in teaching. His research domain is Software Engineering. He has completed B.E in Information Science and Engineering from Kalpataru Institute of Technology, Tiptur, India. And M.Tech in Software Engineering from East Point College of Engineering and Technology, Bangalore, India. He has published 3 research papers in international journals and 5 technical papers in International/National conferences. He is pursuing Ph.D in Computer Science & Engineering from Jain University, Bangalore, India.

Dr. NandaKumar AN, Professor in the Dept of CSE, NHCE, Bangalore. He has more than 35 years of teaching experience. He has completed P.hD from Berhanpur University. He has done BE in Electronics and Communication Engineering from Mysore University, Mysore, India. He has completed M.Tech in Computer Science and Technology from Roorkee University Roorkee. He has published more than 50 research papers in various International journals and international conferences. His area of interests include software Engineering, Image processing, wireless sensor networks, parallel computing and others. He has also served as Principal in many reputed engineering colleges in Karnataka and Andhra including Dean (research) in a reputed university in Tamil Nadu.